AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended)At a computer system including a processor and system memory, [[~~A~~]]a method for determining equivalence of XML schema types, the method being performed by a system configured to generate code from XML schemas, the method comprising:

an act of identifying a first XML schema type, the first XML schema type providing a representation of first schema components;

an act of identifying ~~at least two~~ a second XML schema type[[s]] for which equivalence to the first schema type is to be determined, ~~each of the at least two~~ the second XML schema type[[s]] ~~having at least one~~ providing a representation of second schema components, the representation of second schema components differing from the representation of the first schema components such that presentation of the first schema components and the second schema components is insufficient to determine if the first and second XML schema types are ~~that can be presented differently in~~ equivalent XML schema types;

a step for the processor normalizing ~~each of~~ the ~~identified~~ first XML schema type[[s]] into a normalized first XML schema type in a unified schema format, normalization of the first XML schema type including:

identifying one or more schema components in the first schema components that can be equivalently presented in the unified schema format, the unified schema format differing for the format of both the first and second XML schema types, identifying one or more components in the first schema components including:

identifying any discretionary components in the first schema components that are expressly recited;

identifying any discretionary components in the first schema components that are expressly omitted; and

identifying one or more list of components in the first schema components that are written in a particular order; and

rewriting the first XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the first XML schema type that was that as expressly recited in the first XML schema type and writing a discretionary component to the first XML schema type that was expressly omitted in the first XML schema type and b) rewriting the order of at least one list of components in the first XML schema type into a different predetermined order of the unified format;

a step for the processor normalizing the second XML schema type into a normalized second XML schema type in the unified schema format, normalization of the second XML schema type including:

identifying one or more schema components in the second schema components that can be equivalently presented in the unified schema format, identifying one or more components in the first schema components including:

identifying any discretionary components in the second schema components that are expressly recited;

identifying any discretionary components in the second schema components that are expressly omitted; and

identifying one or more list of components in the second schema components that are written in a particular order; and

rewriting the second XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the second XML schema type that as expressly recited in the second XML schema type and writing a discretionary component to the second XML schema type that was expressly omitted in the second XML schema type and b) rewriting the order of at least one list of components in the second XML schema type into a different predetermined order of the unified format;

a step for determining ~~equivalence of the at least two~~ that the first and second normalized XML schema types are equivalent, including comparing the schema components of the first normalized XML schema type to the schema components of the second normalized XML schema type; and.

an act of creating a single class collectively representing both the first XML schema type and the second XML schema type based on the determination of equivalence and not withstanding that the representation of second schema components differ from the representation of the first schema components, the single class compatible with applications utilizing either the first XML schema type or the second XML schema type.

2. (Currently Amended) The method as recited in claim 1, wherein the step for determining that the first and second XML schema types are equivalent ~~equivalence~~ includes creating and comparing hash numbers of corresponding portions of the first XML schema type and the secone XML schema type~~the at least two normalized XML schema types~~.

Claim 3. (Cancelled).

4. (Currently Amended) The method as recited in claim 1, wherein the step for ~~normalizing each of the identified XML schema types includes writing the at least one schema component in each of the at least two XML schema types according to a unified format and prior to determining equivalence~~ creating a single class collectively representing both the first XML schema type and the second XML schema type comprises an act of creating a class from the normalized first XML schema type.

5. (Currently Amended) The method as recited in claim 4, wherein rewriting the order of at least one list of ~~schema~~ components in the first XML schema type includes altering an order of at least two schema components within ~~a single~~ the first XML schema type.

6. (Previously Presented) The method as recited in claim 5, wherein altering the order includes placing the at least two schema components into alphabetical order.

7. (Currently Amended) The method as recited in claim 5, wherein altering the order includes placing the at least two schema components into numerical order ~~prior to altering the order, it is determined that the order of the at least two schema components is discretionary~~.

8. (Currently Amended) The method as recited in claim [[4]]5, wherein the at least one component is a discretionary component that is not explicitly recited in ~~at least one of~~ the first XML schema type[[s]]~~, and wherein writing the at least one schema component includes writing the at least one schema component for a first time~~.

9. (Currently Amended) The method as recited in claim 1, wherein ~~further including:~~ creating a single class for applications utilizing either the first XML schema type or the second XML schema comprises

~~upon determining equivalence,~~ creating a single class that is used interchangeably by the applications ~~for each equivalent XML schema type~~ so as to enable applications using either the first XML schema or the second XML schema type to share data.

10. (Currently Amended) A computer program product for use at a computer system, the computer program product for implementing a method for determining equivalence of XML schema types, the computer program product comprising one or more computer-readable media having stored thereon computer-executable instructions that, when executed at a processor, cause the computer system to perform the ~~for implementing a~~ method for determining equivalence of XML schema types, ~~the method comprising~~including the following:

identify a first XML schema type, the first XML schema type providing a representation of first schema components;

~~an act of~~ identify[[ing]] ~~at least two~~ a second XML schema type[[s]] for which equivalence to the first schema type is to be determined, ~~each of the at least two~~ the second XML schema type[[s]] ~~having at least one~~ providing a representation of second schema components, the representation of second schema components differing from the representation of the first schema components such that presentation of the first schema components and the second schema components is insufficient to determine if the first and second XML schema types are ~~that can be presented differently in~~ equivalent XML schema types;

~~a step for~~ normalize[[ing]] ~~each of~~ the ~~identified~~ first XML schema type[[s]] into a normalized first XML schema type in a unified schema format, normalization of the first XML schema type including:

identifying one or more schema components in the first schema components that can be equivalently presented in the unified schema format, the unified schema format differing for the format of both the first and second XML schema types, identifying one or more components in the first schema components including:

identifying any discretionary components in the first schema components that are expressly recited;

identifying any discretionary components in the first schema components that are expressly omitted; and

identifying one or more list of components in the first schema components that are written in a particular order; and

rewriting the first XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the first XML schema type

that was that as expressly recited in the first XML schema type and writing a discretionary component to the first XML schema type that was expressly omitted in the first XML schema type and b) rewriting the order of at least one list of components in the first XML schema type into a different predetermined order of the unified format;

normalize the second XML schema type into a normalized second XML schema type in the unified schema format, normalization of the second XML schema type including:

identifying one or more schema components in the second schema components that can be equivalently presented in the unified schema format, identifying one or more components in the first schema components including:

identifying any discretionary components in the second schema components that are expressly recited;

identifying any discretionary components in the second schema components that are expressly omitted; and

identifying one or more list of components in the second schema components that are written in a particular order; and

rewriting the second XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the second XML schema type that as expressly recited in the second XML schema type and writing a discretionary component to the second XML schema type that was expressly omitted in the second XML schema type and b) rewriting the order of at least one list of components in the second XML schema type into a different predetermined order of the unified format;

~~a step for~~ determine[[~~ing~~]] equivalence of the ~~at least two~~ first and second normalized XML schema types, including comparing the schema components of the first normalized XML schema type to the schema components of the second normalized XML schema type; and[[-]]

create a single class collectively representing both the first XML schema type and the second XML schema type based on the determination of equivalence and not withstanding that the representation of second schema components differ from the representation of the

first schema components, the single class compatible with applications utilizing either the first XML schema type or the second XML schema type.

11. (Currently Amended) The computer program product as recited in claim 10, wherein computer-executable instructions that, when executed, cause the computer system to ~~the step for~~ determine[[~~ing~~]] that the first and second XML schema types are equivalent ~~equivalence includes~~ comprise computer-executable instructions that, when executed, cause the computer system to create[[~~ing~~]] and compare[[~~ing~~]] hash numbers of corresponding portions of the first XML schema type and the second XML schema type~~the at least two normalized XML schema types~~.

12. (Currently Amended) The computer program product as recited in claim 10, wherein computer-executable instructions that, when executed, cause the computer system to ~~the step for normalizeing each of the identified XML schema types includes writing the at least one schema component in each of the at least two XML schema types according to a unified format and prior to determining equivalence~~ create a single class collectively representing both the first XML schema type and the second XML schema type comprise computer-executable instructions that, when executed, cause the computer system to create a class from the normalized first XML schema type.

13. (Currently Amended) At a computer system including a processor and system memory, a[[A]] method for determining equivalence of XML schema types in a system configured to generate code from XML schemas, the method comprising:

an act of identifying a first XML schema type, the first XML schema type providing a representation of first schema components;

an act of identifying ~~at least two~~ a second XML schema type[[s]] for which equivalence to the first schema type is to be determined, ~~each of the at least two~~ the second XML schema type[[s]] ~~having at least one~~ providing a representation of second schema components, the representation of second schema components differing from the representation of the first schema components such that presentation of the first schema components and the second schema components is insufficient to determine if the first and second XML schema types are ~~that can be presented differently in~~ equivalent XML schema types;

~~an act of rewriting the at least one schema component in each of at least two XML schema types according to a custom format resulting in corresponding at least two normalized XML schema types;~~

an act of the processor normalizing the first XML schema type into a normalized first XML schema type in a unified schema format, normalization of the first XML schema type including:

identifying one or more schema components in the first schema components that can be equivalently presented in the unified schema format, the unified schema format differing for the format of both the first and second XML schema types, identifying one or more components in the first schema components including:

identifying any discretionary components in the first schema components that are expressly recited;

identifying any discretionary components in the first schema components that are expressly omitted; and

identifying one or more list of components in the first schema components that are written in a particular order; and

rewriting the first XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the first XML schema type

that was that as expressly recited in the first XML schema type and writing a discretionary component to the first XML schema type that was expressly omitted in the first XML schema type and b) rewriting the order of at least one list of components in the first XML schema type into a different predetermined order of the unified format;

an act of the processor normalizing the second XML schema type into a normalized second XML schema type in the unified schema format, normalization of the second XML schema type including:

identifying one or more schema components in the second schema components that can be equivalently presented in the unified schema format, identifying one or more components in the first schema components including:

identifying any discretionary components in the second schema components that are expressly recited;

identifying any discretionary components in the second schema components that are expressly omitted; and

identifying one or more list of components in the second schema components that are written in a particular order; and

rewriting the second XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the second XML schema type that as expressly recited in the second XML schema type and writing a discretionary component to the second XML schema type that was expressly omitted in the second XML schema type and b) rewriting the order of at least one list of components in the second XML schema type into a different predetermined order of the unified format;

~~an act of comparing the at least two normalized XML schema types;~~

an act of generating a first hash number for at least a portion ~~each~~ of the ~~at least two~~ first normalized XML schema type[[s]]; ~~and~~

an act of generating a second hash number for at least a corresponding portion of the second normalized XML schema type; and

an act of the processor determining equivalence of the ~~at least two~~ first and second normalized XML schema types by comparing the first and second hash numbers and wherein

when the first and second hash numbers ~~for each of the at least two normalized XML schema types are the same~~ match then the first and second XML schema types are indicated as being equivalent; and

an act of creating a single class collectively representing both the first XML schema type and the second XML schema type based on the determination of equivalence and not withstanding that the representation of second schema components differ from the representation of the first schema components, the single class compatible with applications utilizing either the first XML schema type or the second XML schema type.

Claim 14. (Cancelled).

15. (Currently Amended) The method as recited in claim 13, wherein rewriting ~~the~~ at least one schema component in the representation of first schema component includes rewriting an existing schema component into a new format.

Claim 16. (Cancelled).

17. (Currently Amended) The method as recited in claim 13, wherein rewriting the order of at least one list of ~~schema~~ components in the first XML schema type includes altering an order of at least two schema components within ~~a single~~ the first XML schema type.

18. (Previously Presented) The method as recited in claim 17, wherein altering the order includes placing the at least two schema components into alphabetical order.

19. (Currently Amended) The method as recited in claim 17, wherein ~~prior to altering the order, it is determined that the order of the at least two schema components is discretionary~~ altering the order includes placing the at least two schema components into numerical order.

20. (Currently Amended) The method as recited in claim 13, ~~further including:~~

~~upon determining equivalence,~~ wherein creating a single class collectively representing both the first XML schema type and the second XML schema type comprises creating a based on

the normalized first XML schema type, the single class ~~that is~~ used interchangeably by applications exchanging data defined in accordance with either the ~~for each equivalent~~ first XML schema type or the second XML schema type.

21. (Currently Amended) The method as recited in claim 13, wherein ~~the~~ at least one of the fist schema components is a schema particle definition.

22. (Currently Amended) The method as recited in claim 13, wherein ~~the~~ at least one of the fist schema components is a schema attribute.

23. (Currently Amended) The method as recited in claim 13, wherein ~~the~~ at least one of the fist schema components is at least one of a child and a sub-child of a named type.

24. (Currently Amended) A computer program product for use at a computer system, the computer program product for implementing a method for determining equivalence of XML schema types, the computer program product comprising one or more computer-readable media having computer-executable instructions that, when executed at a processor, cause the computer system to ~~for~~ implement~~ing a~~ the method for determining equivalence of XML schema types, ~~the method comprising~~ including the following:

identify a first XML schema type, the first XML schema type providing a representation of first schema components;

~~an act of~~ identify~~ing at least two~~ a second XML schema type[[s]] for which equivalence to the first XML schema type is to be determined, ~~each of the at least two~~ the second XML schema type~~s having at least one~~ schema components, the representation of second schema components differing from the representation of the first schema components such that presentation of the first schema components and the second schema components is insufficient to determine if the first and second XML schema types are ~~that can be presented differently in~~ equivalent XML schema types;

~~an act of writing the at least one schema component in each of at least two XML schema types according to a custom format resulting in corresponding at least two normalized schema types;~~

normalize the first XML schema type into a normalized first XML schema type in a unified schema format, normalization of the first XML schema type including:

identifying one or more schema components in the first schema components that can be equivalently presented in the unified schema format, the unified schema format differing for the format of both the first and second XML schema types, identifying one or more components in the first schema components including:

identifying any discretionary components in the first schema components that are expressly recited;

identifying any discretionary components in the first schema components that are expressly omitted; and

identifying one or more list of components in the first schema components that are written in a particular order; and

rewriting the first XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the first XML schema type that was that as expressly recited in the first XML schema type and writing a discretionary component to the first XML schema type that was expressly omitted in the first XML schema type and b) rewriting the order of at least one list of components in the first XML schema type into a different predetermined order of the unified format;

normalize the second XML schema type into a normalized second XML schema type in the unified schema format, normalization of the second XML schema type including:

identifying one or more schema components in the second schema components that can be equivalently presented in the unified schema format, identifying one or more components in the first schema components including:

identifying any discretionary components in the second schema components that are expressly recited;

identifying any discretionary components in the second schema components that are expressly omitted; and

identifying one or more list of components in the second schema components that are written in a particular order; and

rewriting the second XML schema type into the unified schema format, including a) at least one of deleting a discretionary component from the second XML schema type that as expressly recited in the second XML schema type and writing a discretionary component to the second XML schema type that was expressly omitted in the second XML schema type and b) rewriting the order of at least one list of components in the second XML schema type into a different predetermined order of the unified format;

~~an act of comparing the at least two normalized XML schema types;~~

~~an act of~~ generate[[~~ing~~]] a first hash number for at least a portion ~~each~~ of the ~~at least two~~ first normalized XML schema type[[~~s~~]]; ~~and~~

generate a second hash number for at least a corresponding portion of the second normalized XML schema type; and

~~an act of~~ determine[[~~ing~~]] equivalence of the ~~at least two~~ first and second normalized XML schema types by comparing the first and second hash numbers and wherein when the first and second hash numbers ~~for each of the at least two normalized XML schema types are the same~~ match then the first and second XML schema types are indicated as being equivalent; and

create a single class collectively representing both the first XML schema type and the second XML schema type based on the determination of equivalence and not withstanding that the representation of second schema components differ from the representation of the first schema components, the single class compatible with applications utilizing either the first XML schema type or the second XML schema type.

Claims 25-28. (Cancelled).

29. (Currently Amended) The computer program product as recited in claim 28, wherein altering the order of at least one list in the first XML schema type includes placing the at least two schema components into alphabetical order.

30. (Previously Presented) The computer program product as recited in claim 28, wherein prior to altering the order, it is determined that the order of the at least two schema components is discretionary.

Claim 31. (Cancelled).